Attorney Docket: 042390.P12281

APPLICATION FOR UNITED STATES PATENT

FOR

METHOD AND APPARATUS FOR COPY PROTECTING HARDWARE DEVICES

Inventor: Miles J. Penner

Prepared by: Blakely Sokoloff Taylor & Zafman LLP
12400 Wilshire Boulevard, 7th Floor
Los Angeles, California 90025
Phone: (425) 827-8600
Facsimile: (425) 827-5644

CERTIFICATE OF MAILING via EXPRESS MAIL

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR § 1.10 on the date indicated above and addressed to the Assistant Commissioner for Patents, Box Patent Application, Washington, D.C. 20231.

"Express Mail" Label Number EL861981431US Date of Deposit September 28, 2001

Rimma N. Oks

09-28-01
Date

METHOD AND APPARATUS FOR COPY PROTECTING HARDWARE DEVICES

TECHNICAL FIELD

This disclosure relates generally to copy protection of hardware, and in particular, but not exclusively, relates to computer hardware including measures to prevent copying.

BACKGROUND

Software piracy is a common and well-known problem that plagues the computer industry. Less well known and less common, but equally damaging, is the problem of hardware piracy. Hardware piracy does not refer to stealing actual hardware from a manufacturer (that would described simply as "theft"), but rather to unauthorized copying of a hardware designed by a manufacturer. Hardware manufacturers spend substantial amounts of money designing computers and sub-components of computers, such as modems and other network communication devices, video cards, hard-drive controllers, and so on. Hardware piracy costs manufacturers plenty, because unauthorized manufacturers obtain the benefits of the manufacturers investment without any investment of their own. Manufacturers who set the standards for certain components, or whose products are in high demand, are particularly vulnerable to hardware piracy.

Hardware piracy has detrimental effects to both the manufacturer and to the ultimate end-user. For the manufacturer, hardware piracy reduces their profits, deprives them of the benefit of their investment in the development of the particular hardware, and may seriously affect the company's reputation and image if the copied hardware is passed off as having come from the original manufacturer. The consumer suffers because they end up with a product that may or may not perform as well as the original, may adversely affect the performance of their computer or damage their computer. Additionally, if the

copied hardware is passed off as that of a premium manufacturer, the consumer may end up paying an unjustified premium for it.

There are legal remedies for hardware piracy that a manufacturer can pursue in some situations. For example, where a hardware design is obtained as a result of industrial espionage or other theft of a trade secret, the manufacturer can take legal measures to pursue the copier. Unfortunately, however, legal process is slow and remedies are retrospective, so by the time any legal remedy can be had the damage is done and the manufacturer cannot fully recover its loss.

BRIEF DESCRIPTION OF THE DRAWINGS

Non-limiting and non-exhaustive embodiments of the present invention are described with reference to the following figures, wherein like reference numerals refer to like parts throughout the various views unless otherwise specified.

Figure 1 is a drawing of an embodiment of a device of the present invention.

Figure 2 is an embodiment of a data set to be encrypted according to an embodiment of the invention.

Figure 3 is a flowchart illustrating a part of an embodiment of the present invention.

Figure 4 is a flowchart illustrating a second part of the embodiment of the present invention whose first part is shown in Figure 3.

DETAILED DESCRIPTION OF THE ILLUSTRATED EMBODIMENTS

Embodiments of a system and method for hardware copy protection are described herein. In the following description, numerous specific details are described to provide a thorough understanding of embodiments of the invention. One skilled in the relevant art will recognize, however, that the invention can be practiced without one or more of the specific details, or with other methods, components, materials, etc. In other

instances, well-known structures, materials, or operations are not shown or described in detail to avoid obscuring aspects of the invention.

Reference throughout this specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present invention. Thus, the appearances of the phrases "in one embodiment" or "in an embodiment" in various places throughout this specification are not necessarily all referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

Figure 1 illustrates one embodiment of the present invention using an authorized network adapter *A* and an unauthorized network adapter *X*. The unauthorized adapter *X* is so called because it is copy of the network adapter *A* made by an unauthorized manufacturer. The authorized adapter *A* includes, among other things, a memory 10, and an input-output (I/O) chip 12. The memory 10 can be any type of non-volatile memory, such as an Electronic Erasable Programmable Read Only Memory (EEPROM), Erasable Programmable Read Only Memory (EPROM), flash memory, and the like. The I/O chip 12 allows communication via pins 14 between the network adapter *A* and other components in a system, for example a computer or server.

Stored in the memory 12 of the communication adapter *A* there is, among other things, an identification code I_{A1} that uniquely identifies the adapter. For a network adapter conforming to the IEEE 802 standard, the unique code I_{A1} is a Media Access Control (MAC) address, a hardware address that uniquely identifies each node in a network. In any network, each network adapter *A* must have a unique MAC address; otherwise, the network server will be unable to differentiate between the different computers connected to the network. Generally, an authorized manufacturer is allocated a block of MAC numbers for its network adapters. The manufacturer sells its adapters to an Original Equipment Manufacturer (OEM) without an assigned MAC number, and the manufacturer must authorize the OEM to use MAC addresses from the OEM's allocation

of MAC addresses for specific models or groups of network adapters. When the manufacturer sells directly to end users, however, it assigns a MAC address to the card before delivery to the end users. Although described herein in the context of network adapters, the adapters *A* and *X* could equally well be any other kind of device that includes a memory and is identified by some unique identification number or code.

In addition to the unique identification code I_{A1}, the memory 12 has stored thereon an encrypted data set E_A. The encrypted data set E_A results from encrypting an unencrypted data set *M* which includes, among other things, a unique code I_{A2} identical to I_{A1}. Figure 2 illustrates an embodiment of the data set or message *M* for use with a network adapter conforming to the IEEE 802 standard; in other embodiments, however, the message *M* may contain more, less, or different information than that shown. Among other things, the data set or message *M* contains the unique identifier I_{A2}, which will also be the MAC number assigned to the network adapter *A* onto which the encrypted message E_A will be written. Using the operational notation *E(K,M)* to denote the encryption of data set *M* using an encryption key K,

$$I_{A2} \in M \text{ and}$$

$$E_A = E(K,M).$$

In operation of the authorized network adapter *A*, a driver that runs the adapter *A* will first decrypt the encrypted data set E_A, thus returning it to its unencrypted form *M*. Thus, using *D(K, M)* to denote the decryption of data set E_A using encryption key K, the driver performs the operation

$$D(K,E_A) = D(K,E(K,M)) = M .$$

Having obtained the data set *M*, the driver then compares the code I_{A2} found within the data set *M* with the identification code I_{A1} found in the memory 12 or elsewhere on the adapter *A*. If the identification code I_{A2} from the data set M is identical to the code I_{A1} found elsewhere in the memory, then the adapter *A* is authorized, and the driver loads itself onto the network adapter *A* so that it can perform its job. If the above condition is

not met and the identifier I_{A2} does not match the code I_{A1}, then the adapter is an unauthorized adapter X. Should the adapter turn out to be unauthorized, the driver can take various actions, such as notifying a user that they have an unauthorized adapter, or setting the driver so that it will not load itself onto an unauthorized adapter X.

The encryption scheme chosen to encrypt the data set M and decrypt the data set E_A is preferably a statistically strong one; in other words, it should be an encryption scheme that is difficult to break. In one embodiment, the encryption and decryption operations described above are carried out using a public/private encryption scheme. In a public/private encryption scheme, data is encrypted using a private encryption key and is decrypted using a public encryption key. The public key can only be used to decrypt data that was encrypted with the corresponding private key. Examples of public/private encryption schemes include Advanced Encryption Standard (AES), a symmetric, or public/private, algorithm supporting variable length blocks of data. Using a public/private key encryption scheme with a private key K and a public key K_P,

$$E_A = E(K, M) \text{ and}$$

$$M = D(K_P, E_A).$$

In one example using a public/private key encryption scheme the data set M could be encrypted using a manufacturer's private key and the encrypted data set E_A is stored in the memory of the network adapter. The driver that runs the adapter would then use the manufacturer's public key to decrypt the data set M, extract the code I_{A2} and compare it to the code I_{A1} stored on the adapter.

Alternatively, or in addition, multiple levels of encryption could be used to encrypt the data set M. For example, encrypting the data set M twice, once with a first private encryption key K_1 and once with a second private encryption key K_2, the following would be true of the encrypted data set E_A:

$$E_A = E(K_2, E(K_1, M)).$$

Of course, if multiple levels of encryption are used to encrypt the data set M, then multiple levels of decryption would be used to decrypt the data set E_A, and the number of levels of decryption would match the number of levels of encryption. Thus, if K_{P1} and K_{P2} are the public keys corresponding respectively to private keys K_1 and K_2, then

$$M = D(K_{P2}, D(K_{P1}, E_A)).$$

When the network adapter A arrives in the hands of an end user, it is usually as part of a system such as a server, desktop or laptop computer, and will have been handled by at least two parties: the authorized manufacturer of the adapter, and an Original Equipment Manufacturer (OEM) who installs the adapter A in the larger system, and then sells it to the end user. In one embodiment, the data set M is encrypted twice—once with a private key of the manufacturer, and once with a private key of the OEM. The driver that operates the adapter then decrypts the data set M using the public keys of both the OEM and the manufacturer. In this way, responsibility for preventing hardware copying is shared by both the manufacturer and the OEMs to which it sells its adapters. Other embodiments using different combinations of keys are also possible. For example, in a case where the manufacturer sells directly to end user (*i.e.*, there is no OEM in the supply chain), then the manufacturer could encrypt the data set M twice using two different private keys of its own. Similarly, in a case where there are more than two parties in the supply chain, the data set could also be encrypted more than twice, using private keys of each party in the supply chain. Responsibility for preventing hardware theft would thus be shared by all parties in the supply chain.

Also shown in Figure 1 is an unauthorized network adapter X. The adapter X will have been assigned an identification code I_{X1} by the unauthorized manufacturer, and the code I_{X1} will be stored in the memory. In the case of a network adapter conforming to the IEEE 802 standard, the code I_{X1} is the MAC numbers assigned to the adapter X by the unauthorized manufacturer. The unauthorized manufacturer can attempt to create a data set M including an identifier I_{X2}, encrypt it into an encrypted data set E_X, and store it on the memory. The unauthorized manufacturer, however, does not possess

the proper encryption key necessary to encrypt the data *M*. Because the data set M will not have been encrypted with the proper encryption key, the driver will either be unable to decrypt the data set E_X or will decrypt it in such a way that the code I_{X2} contained in the encrypted data *M* will not match the code I_{X1} assigned to the adapter *X* by the unauthorized manufacturer. The driver with thus determine that the adapter is, in fact, an unauthorized adapter *X*. In such a case, the driver can take various actions, such as notifying a user that they have an unauthorized adapter, or setting the driver so that it will not load itself onto an unauthorized adapter *X*.

One way an unauthorized manufacturer could circumvent this embodiment is by making an exact copy of the *entire* memory of an adapter A, and then transferring that exact copy to the memory of adapter *X*; in such a case, both the unencrypted code I_{A1} and the encrypted data set E_A containing the code I_{A2} are copied onto the unauthorized adapter *X*, such that $I_{X1} = I_{A1}$ and $E_X = E_A$, such that $I_{X2} = I_{A2}$. Upon performing the decryption described above, the driver would find that the identifier I_{X2} matches the code I_{XI}, and would conclude that the adapter is an authorized one. Such a scheme would not be practical for an unauthorized manufacturer, however, because it would either have to copy the memory from one adapter A to many adapters *X*, such that all adapters *X* would have the same identification code I_{X1}, or it would have to obtain one authorized adapter *A* for every adapter *X* it wished to produce, so that the adapters *X* would not have duplicate identification code. Fortunately, neither of these options is feasible for an unauthorized manufacturer: the first would certainly lead to customer complaints, particularly for large customers who network many computers, and the second would be too expensive.

Figure 3 illustrates an embodiment of a process 20 by which an authorized manufacturer and an OEM cooperate to prevent unauthorized copying of adapters. The dashed line in the figure indicates the delineation between tasks performed by the OEM and tasks performed by the authorized manufacturer. Although shown as a process including the participation of only two parties (OEM and manufacturer), the process could also take place with more or less parties involved.

Beginning at 22, the OEM submits for approval by the manufacturer network adapter information and Ethernet or MAC addresses which it proposes to use for its adapters. The information submitted by the OEM for each network adapter is contained in a data set or message *M*. At 24, the manufacturer receives the message *M* and determines whether the request from the OEM is valid—that is, whether the request comes from an OEM to whom legitimate sales have been made, and whether it requests valid MAC or Ethernet addresses. If the request is not valid, the manufacturer notifies the OEM at 26 and informs the OEM of the request's invalidity, and the OEM must then submit new information to the manufacturer for approval. If the request is valid, then at 28 the manufacturer approves the data set or message *M* submitted by the OEM. Once approved, the manufacturer "signs" the data set or message *M* by encrypting it with its own private key K_p, resulting in an encrypted data set or message $E(K_p,M)$. At 30, the manufacturer sends the encrypted message *M* to the OEM, which encrypts the encrypted message $E(K_p,M)$ with its own private key K_{op}, resulting in a twice-encrypted message $E(K_{op},E(K_p,M))$. Finally, at 32, the twice-encrypted message $E(K_{op},E(K_p,M))$ is written onto the memory 10 of the network adapter *A*, in this case to the EEPROM of the adapter *A*.

Figure 4 illustrates an embodiment of a process by which a driver that operates the network adapter determines whether the adapter is an authorized adapter A or an unauthorized adapter *X*. At 34, the driver loads into the network adapter and reads the encrypted data set or message E from the memory 10, which can be any kind of non-volatile memory such as EEPROM. As described above, the encrypted message E results from two levels of encryption, such that $E= E(K_{op},E(K_p,M))$.

Once the device driver loads the encrypted data set or message E from the EEPROM, at 36 it uses the public keys of the OEM the manufacturer to decrypt the message E, which was previously encrypted using the private keys of the manufacturer and the OEM. The decryption takes place in the reverse order of the encryption; that is, the message is first decrypted using the public key K_{oc} of the OEM, and then decrypted

using the public key K_M of the manufacturer. The resulting decrypted message $D(K_{oc},D(K_m,E))$ is thus the result of the two levels of decryption. At 38, the driver compares the unencrypted MAC address stored in the memory with the MAC address found in the decrypted message to determine if the network adapter is authorized or unauthorized. If the decrypted MAC address does not match the unencrypted MAC address found in the memory, this means that the MAC address is not valid for the adapter. The driver can then take various actions, such as notifying a user that they have an unauthorized adapter, or refusing to load itself onto the unauthorized adapter *X*. If the MAC address from the decrypted message does match the unencrypted MAC address found on the memory, then the MAC address is valid for the adapter, the adapter is an authorized one, and the driver proceeds with normal loading an operation. Of course, different encryption schemes can be used in the present invention. For example, the original message *M* may be encrypted more or less then twice as shown, or may be encrypted using different varieties of keys, and using different encryption algorithms, such as AES or 3DES.

The above description of illustrated embodiments of the invention, including what is described in the Abstract, is not intended to be exhaustive or to limit the invention to the precise forms disclosed. While specific embodiments of, and examples for, the invention are described herein for illustrative purposes, various equivalent modifications are possible within the scope of the invention, as those skilled in the relevant art will recognize.

These modifications can be made to the invention in light of the above detailed description. The terms used in the following claims should not be construed to limit the invention to the specific embodiments disclosed in the specification and the claims. Rather, the scope of the invention is to be determined entirely by the following claims, which are to be construed in accordance with established doctrines of claim interpretation.